Jurnal Metris, 14 (2013): 113 – 120

Jurnal
Metris
ISSN: 1411 - 3287

Perbandingan Algoritma Bee Colony dengan Algoritma Bee Colony Tabu List dalam Penjadwalan Flow Shop

Andre Sugioko

Department of Industrial Engineering, Faculty of Engineering
Universitas Katolik Atma Jaya – Jakarta
Jl. Jendral Sudirman 51, Jakarta 12930
Telepon (021) 5727615
E-mail : andresugioko@yahoo.com

Abstract.

This paper discusses the performance of Bee Colony algorithm integrated with the swap method and tabu list with Bee Colony algorithm. This research was based on the results of research by Chong in 2006, which declared that the Bee Colony algorithm integrated with the swap method has a good performance, but cannot compete with the performance of Tabu Search, so the researchers will test whether integrating Bee Colony algorithm with swap method and tabu list will give better performance than Bee Colony algorithm or not. This research will use two study case flow shop scheduling problem, with objective function is minimum makespan to compare both algorithm's performance. The results Bee Colony-Tabu provide superior results at makespan values than Bee Colony algorithm, but not at computational time.

Keyword: Scheduling, Bee Colony, Tabu List, Flow Shop, Makespan

1. PENDAHULUAN

Optimasi adalah salah satu disiplin ilmu dalam matematika yang fokus untuk mendapatkan nilai minimum atau maksimum secara sistematis dari suatu fungsi, peluang maupun pencarian nilai lainnya dalam berbagai kasus. Optimasi sangat berguna di hampir segala bidang dalam rangka melakukan usaha secara efektif dan efisien untuk mencapai target hasil yang ingin dicapai, sehingga Optimasi sangat penting dalam persaingan di dunia industri yang sudah sangat ketat di segala bidang yang ada. Di antaranya optimasi penjadwalan produksi adalah permasalahan yang paling banyak dianalisa dimana penjadwalan produk ada yang berupa *job shop* dan *flow shop*, keduanya banyak diselesaikan dengan pendekatan metaheuristik. Salah satu pendekatan metaheuristik adalah Algoritma *Bee Colony*.

Lebah merupakan serangga yang terpadu. Kemampuan bertahan hidup seluruh koloni lebah tergantung dari tiap individu lebah. Lebah menggunakan tugas – tugas yang sistematik yang diantaranya bertujuan menjaga eksistensi koloninya. Mereka melakukan bermacan – macan pekerjaan seperti *forgaging*, reproduksi, membangun sarang, dan lainnya. Dari pekerjaan – pekerjaan ini, *forgaging* merupakan aktivitas yang penting untuk menjaga pasokan makanan pada sarang lebah.

Kebiasaan lebah diadaptasi menjadi algoritma untuk memecahkan permasalahan yang kompleks. Beberapa diantaranya seperti alokasi server dinamik (Nakrani and Tovey, 2004), optimalisasi, *routing* jaringan telekomunikasi (Karaboga and Basturk, 2008) permasalahan stokastik akan *vehicle routing* (Luˇ ci´ c and Teodorovi´ c, 2003), dan *job shop scheduling* (Chong, et al, 2006 dan 2007), yang sering disebut dengan *Bee Colony Algorithm*.

Bee Colony Algorithm merupakan salah satu algoritma dari algoritma metode *metaheuristik*. Metode *metaheuristik* menggunakan pencarian secara acak, dan dapat digunakan untuk rentang permasalahan yang lebih luas, namun hasil tidak selalu mencapai *global optimum*. Metode *metaheuristik* bergantung akan prosedur pembantu seperti mutasi, dan batasan-batasan, sehingga prosedur pembantu inilah yang menentukan baik-buruknya performa dan lama penyelesaian dari suatu algoritma metode *metaheuristik*.

Dalam penelitian ini, penulis menganalisa performa algoritma *Bee Colony* yang telah dimodifikasi dengan menggunakan operasi *swap* dan *tabu list* yang dalam penelitian Chong pada tahun 2006, yang menyatakan bahwa pencarian

solusi dengan menggunakan operasi *swap* akan memberikan hasil yang lebih baik daripada pencarian solusi terencana seperti dalam penelitian Chong tahun 2006, dan dengan menggunakan *tabu list* akan menolong untuk lepas dari *local optimum*. Masalah yang akan digunakan dalam melihat performa algoritma *Bee Colony* menggunakan masalah penjadwalan *flow shop* dengan fungsi tujuan minimasi total waktu produksi.

2. METODOLOGI

Menurut Morton dan Pentico (1993), "penjadwalan adalah suatu proses pengaturan, pemilihan dan penentuan waktu sumber daya yang berguna untuk menyelesaikan semua aktivitas yang diperlukan untuk memproduksi output yang diinginkan pada waktu yang diharapkan, dimana juga terdapat kendala-kendala diantara aktivitas-aktivitas dan sumber daya yang ada"

Penjadwalan juga didefinisikan sebagai proses pengurutan pembuatan produk secara menyeluruh pada sejumlah mesin tertentu. Penjadwalan juga dipandang sebagai proses pengalokasian sumber untuk memilih tugas dalam jangka waktu panjang. Penjadwalan merupakan alat ukur yang baik bagi perencana agregat. Pesanan-pesanan aktual pada tahap ini akan ditugaskan pertama kalinya pada sumber daya tertentu (fasilitas, pekerja dan peralatan), kemudian dilakukan pengurutan kerja pada tiap-tiap pusat pemrosesan sehingga dicapai optimalitas utilisasi kapasitas yang ada. (Arman, 2003).

2.1. Penjadwalan *Flow Shop*

Proses produksi dengan aliran *flow shop* berarti proses produksi dengan pola aliran identik dari satu mesin ke mesin lain. Walaupun pada *flow shop* semua tugas akan mengalir pada jalur produksi yang sama, yang biasa dikenal sebagai pure *flow shop*, tetapi dapat pula berbeda dalam dua hal. Pertama, jika *flow shop* dapat menangani tugas yang bervariasi. Kedua, jika tugas yang datang ke dalam *flow shop* tidak harus dikerjakan pada semua jenis mesin. Jenis *flow shop* seperti ini disebut *general flow shop.*

Karakteristik penjadwalan *job Shop* dapat dijabarkan sebagai berikut:

- Ada sejumlah m mesin dan sejumlah *n job*.
- Setiap job terdiri dari satu rantai urutan yang serupa satu sama lain.
- Setiap operasi dalam job diproses oleh salah satu mesin yang ada dengan waktu proses yang diasumsikan tetap.
- Permasalahan penjadwalan untuk model *job Shop* merupakan salah satu permasalahan optimasi kombinatorial yang kompleks sehingga disebut *NP-hard* (*NP* merupakan singkatan dari *nondeterministic polynomial*).

2.2 Algoritma *Bee Colony*

Koloni Lebah

Koloni dari lebah mampu menempuh jarak yang cukup jauh (lebih dari 10 km) dan mampu untuk bergerak ke segala arah secara simultan untuk memeriksa lebih dari satu sumber makanan (Von Frisch K, 1976, Seeley TD, 1996). Koloni ini bekerja dengan mengirimkan lebah pengintainya ke ladang yang banyak sumber makanannya. Proses eksplorasi (*forgaging*) dimulai dari mengirimkan lebah pengintai untuk mencari bunga yang berpeluang memiliki madu yang banyak, lebah pengintai tersebut bergerak secara acak dari satu tangkai bunga ke tangkai bunga yang lainnya. Pada saat musim panen, koloni tetap melakukan ekplorasinya, dan tetap mempertahankan populasi dari lebah pengintai. (Seeley TD, 1996). Ketika lebah pengintai kembali ke sarang dan menemukan bunga dengan kadar gula/madu yang dianggap cukup tinggi daripada yang diharapkan, akan mengambil nektarnya sebagai sampel lalu melakukan tarian untuk memberikan lokasi bunga tersebut, yang disebut dengan "*waggle dance*". (Von Frisch K, 1976).

Tarian ini sangat penting bagi komunikasi dalam koloni, dan menyimpan akan tiga informasi mengenai bunga yang lebah tersebut temukan, yaitu: arah di mana dapat menemukan bunga tersebut, jarak yang harus ditempuh dari sarang lebah ke bunga, dan kualitas akan madunya (Von Frisch K, 1976). Informasi ini membantu koloni untuk mengirimkan lebah yang lainnya ke bunga tersebut tanpa menggunakan petunjuk atau peta. Tiap lebah memiliki pengetahuan akan lingkungan sekitarnya dari "*waggle dance*" serta terdapat *pheromone* yang juga membantu para lebah untuk mencari lokasi bunga tersebut. Dan tarian ini juga digunakan untuk mengevaluasi keuntungan dari bunga yang lainnya, yang berdasarkan energi yang diperlukan, dan jumlah hasil yang mereka bisa dapatkan (Camazine S, et al, 2003). Setelah melakukan tarian, lebah pengintai tersebut akan kembali ke bunga yang telah ditemukan dan diikuti oleh beberapa lebah lainnya, hal ini diperuntukkan untuk mencari bunga berkualitas yang lainnya. Sehingga dengan ini koloni mampu untuk mengumpulkan makanan dengan cepat dan efisien.

Algoritma *Bee Colony*

Seperti yang telah dijabarkan pada bagian atas, algoritma lebah ini merupakan algoritma untuk optimalisasi yang terinspirasi dari kebiasaan eksplorasi lebah (*forgaging*) untuk mencari solusi optimal. Proses algoritma Bee Colony Optimalisasi

secara umum dibagi dalam beberapa tahap, yaitu: (Pham D.T., et al, 2006).

1. Melakukan sebanyak n pencarian terhadap area solusi yang telah ditentukan. Hal ini menunjukan bahwa dicari sebanyak n solusi dari sedemikian banyaknya solusi, yang kemudian akan diuji.
2. Tiap calon solusi akan diuji performansinya dengan menggunakan *fitness test*.
3. Solusi yang memiliki nilai *fitness* tinggi akan dipilih untuk dilakukan *neighbourhood search*. Yaitu melakukan pencarian solusi dari solusi yang telah dipilih untuk didapatkan solusi baru tapi masih berasal dari solusi awal. Yang selalu berubah-ubah sesuai fungsi tujuan hingga mencapai nilai optimum.
4. Melakukan *neighbourhood search* pada kumpulan solusi yang tidak terpilih. Dan dilakukan uji *fitness* untuk menguji performansinya. Pengujian *fitness* akan menentukan apakah solusi tersebut akan terpilih kembali oleh lebah yang lainnya atau tidak (dalam bentuk persentase).
5. Dilakukan berulang hingga kriteria berhenti tercapai. Dan dipilih yang memiliki nilai *fitness* tertinggi.

2.3 Algoritma *TABU SEARCH* (TS)

Tabu search merupakan suatu metode optimasi matematis yang termasuk ke dalam kelas *local search*. TS memperbaiki performansi *local search* dengan memanfaatkan penggunaan struktur memori. TS diperkenalkan pertama kali oleh Glover (Glover, 1986), dengan ide dasar disampaikan oleh Hansen (Hansen, 1986).

Tabu List

Struktur memori fundamental dalam *tabu search* dinamakan *tabu list*. *Tabu list* menyimpan atribut dari sebagian *move* (transisi solusi) yang telah diterapkan pada iterasi-iterasi sebelumnya. *Tabu search* menggunakan *tabu list* untuk menolak solusi-solusi yang memenuhi atribut tertentu guna mencegah proses pencarian mengalami *cycling* pada daerah solusi yang sama, dan menuntun proses pencarian menelusuri daerah solusi yang belum dikunjungi. Tanpa menggunakan strategi ini, *local search* yang sudah menemukan solusi optimum local dapat terjebak pada daerah solusi optimum local tersebut pada iterasi-iterasi berikutnya. *List* ini mengikuti aturan LIFO dan biasanya sangat pendek (panjangnya biasanya sebesar $O(\sqrt{N})$, dimana N adalah jumlah total dari operasi). Setiap saat ada langkah itu akan ditempatkan dalam *tabu list Tabu list* hanya menyimpan langkah transisi (*move*) yang merupakan lawan atau kebalikan dari langkah yang telah digunakan dalam iterasi sebelumnya untuk bergerak dari satu solusi ke solusi berikutnya. Dengan kata lain *tabu list* berisi langkah-langkah yang membalikan solusi yang baru ke solusi yang lama (Glover, E. Et al, 1993).

Pada tiap iterasi, dipilih solusi baru yang merupakan solusi terbaik dalam *neighbourhood* dan tidak tergolong sebagai tabu. Kualitas solusi baru ini tidak harus lebih baik dari kualitas solusi sekarang. Apabila solusi baru ini memiliki nilai fungsi objektif lebih baik dibandingkan solusi terbaik yang telah dicapai sebelumnya, maka solusi baru ini dicatat sebagai solusi terbaik yang baru. Apabila terdapat *move* yang dinilai dapat menghasilkan solusi yang dinilai dapat menghasilkan solusi yang baik namun *move* tersebut berstatus tabu, maka *move* tersebut dapat digunakan untuk membentuk solusi berikutnya (status tabunya dibatalkan). Hal ini merupakan kondisi khusus pada *tabu list* yang dikenal dengan kriteria aspirasi atau kondisi aspirasi.

2.4 Modifikasi Algoritma *Bee Colony*

Tahap *Forgaging*

Mengacu penelitian Chong tahun 2006, yang menyatakan bahwa dengan menggunakan pencarian solusi metode *swap* akan memberikan hasil yang lebih baik daripada pencarian terencana, sehingga dalam penelitian ini pencarian solusi pertama akan menggunakan pencarian acak dan pada pencarian solusi alternative akan digunakan pencarian metode *swap*.

Tahap Seleksi Atau *Waggle Dance*

Penelitian ini akan menggunakan pemilihan solusi sebanyak *k*, dengan kriteria nilai *makespan* terpendek. Sebab penggunaan aturan diatas membantu untuk memastikan bahwa solusi akhir yang akan terpilih merupakan solusi yang lebih unggul daripada seluruh *makespan* yang telah dibentuk. Program yang dibuat akan menggunakan nilai *makespan* yang terendah sebagai solusi terbaik. Langkah algoritma *bee colony* yang digunakan adalah

1. Inisialisasi solusi awal
2. *Forgaging* 1: cari *n* (populasi lebah)solusi baru dengan metode *Swap*, dari solusi awal
3. *Waggle Dance*:
   a. Pilih *k*-solusi terbaik sebanyak ukuran solusi
   b. Update list solusi
   c. Update *Tabu List*
4. *Forgaging* 2: cari *n* (populasi lebah)solusi baru dengan metode *Swap*, dari seleksi solusi terbaik secara acak.
5. Iterasi : n +1
6. Lakukan hingga Iterasi N
7. Selesai

Solusi yang dipilih pada tahap ini yang memiliki waktu *makespan* terendah dan solusi yang terpilih akan dimasukan kedalam *tabu list*. *Tabu list* ini diujukan untuk mengurangi kemungkinan terjadinya pencarian solusi secara local.

3. HASIL DAN PEMBAHASAN

3.1. Pengujian Modifikasi Algoritma *Bee Colony*

Pada bagian ini akan disajikan hasil dari penggunaan algoritma *Bee Colony*. Dimana hasil pengolahan ini menggunakan permasalahan penjadwalan *flow shop*. Dimana variasi permasalahan terdapat pada jumlah pekerjaan yang harus diselesaikan, yaitu 100, dan 200 pekerjaan.

Pengujian akan dilakukan dengan menggunakan nilai-nilai parameter pada tabel 1, dimana dengan menggunakan nilai parameter yang serupa diharapkan perbedaan antara algoritma dapat terlihat jelas. Nilai parameter ini diambil berdasarkan beberapa pertimbangan. Yaitu untuk panjang *tabu list* tujuh (7) berdasarkan saran Glover pada tahun 1993 untuk permasalahan penjadwalan , jumlah solusi tetangga yang bernilai lima (5). Program yang digunakan dalam penelitian ini adalah program MATLAB. Dimana program dijalankan dengan spesifikasi komputer, yaitu Intel(R) Core(TM)2 Quad CPU Q9550 @2.83Ghz, 4GB of RAM, Sistem Operasi: Microsoft Windows XP. Pengujian dilakukan direplikasi sebanyak 3 kali, dengan mencatat akan nilai makespan dan waktu lamanya permasalahan diproses. Hasil yang didapat disajikan pada tabel 2.

Tabel 1. Parameter Pengujian dan Evaluasi Performa Algoritma

Parameter	*Tabu Search*	*Bee Colony*	*Bee Colony-Tabu*
Jumlah solusi tetangga	5	5	5
Panjang *tabu list*	7	-	7
Jumlah iterasi	10	10	10
Panjang List	-	5	7

Tabel 2. Hasil Performa Algoritma *Tabu Search, Bee Colony, Bee Colony-tabu list*

Jumlah *Job*	100 *Job*	200 *Job*
Tabu Search (TS)		
Makespan (Menit)	21298,333	28195
Waktu Proses Komputer (Detik)	49715	64316
Bee Colony		
Makespan (Menit)	21215	28115
Waktu Proses Komputer (Detik)	49330,667	62371,667
Bee Colony-Tabu		
Makespan (Menit)	21215	28115
Waktu Proses Komputer (Detik)	34771	47806

3.2. Aplikasi Algoritma *Bee Colony*

Pada bagian ini akan algoritma *Bee Colony* akan diujikan pada permasalahan nyata dengan menggunakan data dari dua perusahaan yang berbeda, dimana satu perusahaan merupakan perusahaan yang bergerak dibidang pembuatan *Non Woven* dan perusahaan yang bergerak dibidang pembuatan *Hydraulic Excavator*, namun yang akan dibahas hanya untuk *stage* fabrikasi dan hanya pada komponen *Revo Frame*.

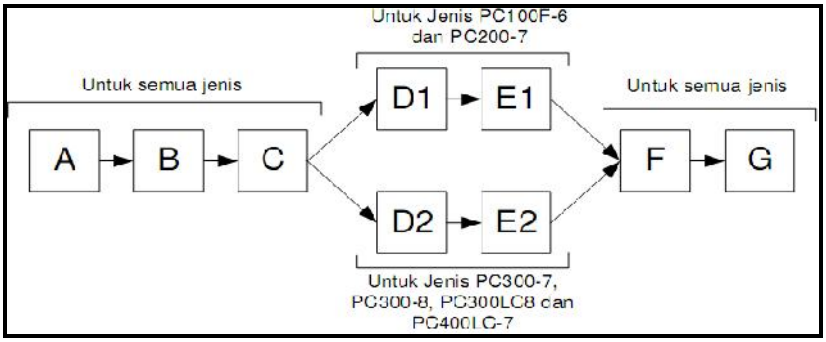
Untuk perusahaan *Non Woven*, akan dibahas lini yang memproduksi jenis produk untuk *Non Woven* Sepatu adalah *Chemi Sheet* 44”, *Chemi Sheet* 54”, Primabond 44”, dan Primabond 54”. Sedangkan untuk *Polyester Sheet* adalah AP850, *Geotext*, dan *Glass Woll*. Perusahaan ini menggunakan sistem *make to* stock untuk perencanaan produksinya, dan di tambah dengan keterbatasan mesin yang membuat perusahaan memberlakukan kebijakan untuk memproduksi satu jenis produk hingga target terpenuhi kemudian melanjutkannya ke produk lainnya. Pada tabel 3 dan 4, merupakan hasil perhitungan waktu baku yang telah dilakukan untuk setiap produk yang ada. Perusahaan *revo frame*, memiliki permasalahan yang menarik dimana terdapat lini yang merupakan mesin yang sama namun dipakai untuk produk tertentu saja yang dapat dilihat pada gambar 1, dan perusahan ini merencanakan produksi dapat dilihat pada tabel 5 yang telah disatukan dengan waktu prosesnya.

Tabel 3. Hasil Perhitungan Total Waktu Siklus untuk Memenuhi Rencana Produksi Lini 1 (Satuan : Menit)

Jenis Produk	Jumlah unit (Roll)	Waktu Siklus Lini 1	Total Waktu
Chemi sheet 44 "& 0,8"	40	6.126	245.04
Chemi sheet 44 "& 1,0"	85	7.66	651.1
Chemi sheet 54 "& 0,8"	15	6.65	99.75
Chemi sheet 54 "& 1,0"	25	8.19	204.75
Primabond 44" & 0,8"	85	6.126	520.71
Primabond 44" & 1,0"	125	7.66	957.5
Primabond 54" & 0,8"	10	6.65	66.5
Primabond 54" & 1,0"	20	8.19	163.8
AP 850	100	9.77	977
Geotex	15	9.77	146.55
Glass woll	10	9.77	97.7

Tabel 4. Hasil Perhitungan Total Waktu Siklus untuk Memenuhi Rencana Produksi Lini 2 (Satuan : Menit)

Jenis Produk	Jumlah unit (Roll)	Waktu Siklus Lini 2	Total Waktu
Chemi sheet 44 "& 0,8"	40	13.385	535.4
Chemi sheet 44 "& 1,0"	85	16.36	1390.6
Chemi sheet 54 "& 0,8"	15	14.83	222.45
Chemi sheet 54 "& 1,0"	25	17.58	439.5
Primabond 44" & 0,8"	85	13.385	1137.725
Primabond 44" & 1,0"	125	16.36	2045
Primabond 54" & 0,8"	10	14.83	148.3
Primabond 54" & 1,0"	20	17.58	351.6



Gambar 1 Rute Operasi *Revo Frame*

Tabel 5. Waktu Operasi *Revo Frame* dan Target Produksi

Jenis Revo Frame	Waktu Proses (Menit)									Target
	A	B	C	D1	D2	E1	E2	F	G	
PC 100F-6	50	140	15	150	0	120	0	90	155	10
PC 200-7	60	140	15	210	0	240	0	150	155	15
PC 300-7	120	140	15	0	210	0	300	160	180	22
PC 300-8	120	140	15	0	210	0	300	160	180	9
PC 300LC-8	120	140	15	0	210	0	300	160	180	21
PC 400LC-7	120	150	15	0	210	0	340	150	200	13

3.3. Model Matematis

Berdasarkan permasalahan yang dipakai, maka model matematis yang dipakai dalam penelitian ini adalah model matematis untuk penjadwalan *flow shop*, dengan fungsi tujuan *minimum makespan*, dengan keterbatasan setiap mesin tidak dapat memproses dua *job* secara bersamaan dan kendala urutan *job*.
Keterangan Notasi:

t = waktu mulai operasi

t_{ij} = waktu mulai operasi j pada *job* i

$^{2}_{ij}$ = waktu proses operasi j pada *job* i

t_{aj} = waktu mulai operasi j pada *job* a

$^{2}_{aj}$ = waktu proses operasi j pada *job* a

t_{bj} = waktu mulai operasi j pada *job* b

$^{2}_{bj}$ = waktu proses operasi j pada *job* b

T_i = Waktu akhir *Job* i
y_{abj} = apabila *job* a datang dahulu maka y_{abj} bernilai 0,
apabila *job* b datang dahulu maka y_{abj} bernilai 1

Fungsi Tujuan:
Meminimalkan T
$T = \text{Max}(T_i)$ ..... (1)
Dimana:
$T_i = t_{ij} + {}^{2}_{ij}$

Batasan-batasan:
Kendala *Job* yang berurutan
$t_{i(j+1)} - t_{ij} \geq {}^{2}_{ij}$ ..... (2)
Kendala satu mesin tidak dapat memproses dua *job* secara bersamaan
$t_{aj} \geq t_{bj} + {}^{2}_{bj}$ Atau $t_{aj} - t_{bj} \geq {}^{2}_{bj}$ untuk *job b* datang pertama $\forall b, \forall a, \forall j$
atau
$t_{bj} \geq t_{aj} + {}^{2}_{aj}$ Atau $t_b - t_{aj} \geq {}^{2}_{aj}$ untuk *job a* datang pertama $\forall a, \forall b, \forall j$
Kedua bentuk diatas bukan bentuk linier, sehingga harus diubah menjadi bentuk linier seperti pada bagian berikut:
$M(y_{abj}) + t_{aj} - t_{bj} \geq {}^{2}_{bj}$ ..... (3)
$M(1 - y_{abj}) + t_b - t_{aj} \geq {}^{2}_{aj}$ ..... (4)
Dimana, M : Bilangan sangat besar.

y_{abj} { 1 (b setelah a); 0 (a setelah b) }

Variabel keputusan:

$t_{ij} \geq 0$ ..... (5)
$y_{abj} = 1,0$

3.4 Pendekatan Algoritma Bee Colony pada Penjadwalan

Lebah
Setiap lebah dianggap sebagai urutan *job*. Lebah yang memiliki nilai *makespan* terendah akan dijadikan acuan untuk lebah – lebah yang lain, untuk mencari solusi yang lebih baik. Nilai *makespan* didapatkan dari waktu pekerjaan terakhir selesai diproses oleh mesin akhir.

Forgaging
Lebah awal akan dijadikan acuan untuk lebah lainnya untuk didapatkan sejumlah solusi (*Forgaging* 1) dengan menggunakan metode *swap*, hal ini bertujuan untuk mendapatkan sejumlah solusi. Solusi yang didapat berikutnya dipilih salah satu secara acak sebagai acuan lebah – lebah yang baru (*Forgaging* 2) dimana, akan dibangkitkan kembali sebanyak *n* solusi baru dengan metode *swap*.

Waggle Dance
Lebah – lebah akan diseleksi dengan memilih nilai *makespan* terendah, kemudian dilakukan pengurutan lebah dari nilai *makespan* terendah hingga tertinggi sesuai dengan batas jumlah solusi terbaik (panjang list solusi). Solusi – solusi yang telah didapatkan akan direkam oleh *tabu list*, untuk mencegah terjadinya pengulangan solusi atau mencegah terjadinya *local optimum*.

3.5 Hasil Perhitungan

Berikut hasil perhitungan dengan menggunakan algoritma *Bee Colony* untuk perusahaan *Non Woven* disajikan pada tabel 6, dan dengan hasil penjadwalan perusahaan. Dan tabel 7 untuk perusahaan *Revo Frame*.

Tabel 6. Perbandingan Solusi Algoritma *Bee colony* dengan Metode Perusahaan *Non Woven*

Solusi	*Makespan* (Menit)	Waktu Komputasi (Menit)
Bee Colony	6270,575	1
Bee Colony - Tabu	6270,575	1
Perusahaan	7095,125	

Tabel 7. Perbandingan Solusi Algoritma *Bee colony* dengan Metode Perusahaan *Revo Frame*

Solusi	*Makespan* (Menit)	Waktu Komputasi (Menit)
Bee Colony	20882,5	1,7925
Bee Colony – Tabu	20845	1,807
Perusahaan	20905	

Dari Tabel 2 terlihat bahwa algoritma *Bee Colony* dan *Bee Colony-Tabu* unggul untuk pencarian nilai m*akespan* dan waktu perhitungan yang lebih kecil daripada *Tabu Search*, namun untuk waktu perhitungan hanya algoritma *Bee Colony* yang lebih unggul daripada Tabu Search dan *Bee Colony-Tabu*. Nilai *makespan* yang lebih rendah daripada Tabu Search dikarenakan algoritma *Bee Colony* dan *Bee Colony-Tabu* menggunakan banyak solusi terbaik yang telah direkam dalam setiap iterasinya, untuk dicari solusi alternatifnya sedangkan algoritma TS hanya mencari alternatif solusi dari satu solusi terbaik. Dengan demikian dengan melakukan pemilihan dari sejumlah solusi terbaik yang ada pada algoritma *Bee Colony* dan *Bee Colony-Tabu*, akan meningkatkan kemungkinan didapatnya solusi yang mendekati optimum.

Pada tabel 6 dan 7, terlihat bahwa algoritma *Bee Colony-Tabu* memiliki hasil yang lebih baik daripada algoritma *Bee Colony* dan metode perusahaan. Metode yang digunakan perusahaan menggunakan prinsip mengerjakan produk yang memiliki waktu proses yang lama, namun tidak memperhitungkan alur proses produksinya sehingga hasil yang didapat kurang baik. Nilai waktu perhitungan untuk algoritma yang menggunakan *tabu list* terlihat lebih lama daripada yang tidak menggunakannya. Hal ini dikarenakan program memerlukan waktu untuk menguji setiap solusi alternatif apakan solusi tersebut ada urutan pekerjaan yang telah dilarang dalam tabu list atau tidak, banyaknya jumlah pekerjaan yang diujikan dan jumlah iterasi, maka hal tersebut mengakumulasi waktu perhitungan menjadi lebih lama daripada algoritma yang tidak menggunakan *tabu search*. Namun dengan adanya *tabu list*, alternatif solusi yang didapatkan merupakan solusi yang berbeda daripada solusi awal yang dipakai untuk pencarian solusi alternatif atau solusi tetangga.

4. KESIMPULAN DAN SARAN

Berdasarkan hasil pengujian algoritma *Bee Colony* dalam penjadwalan *flow-shop*. Algoritma *Bee Colony* dan *Bee Colony-Tabu* memberikan hasil yang unggul untuk pencarian nilai m*akespan* dan waktu perhitungan yang lebih kecil daripada *Tabu Search*. Dimana keunggulan ini terletak pada pencarian solusi alternatif pada proses *waggle dance*, yang mampu untuk meningkatkan kemungkian didapatnya solusi yang mendekati optimum.

Walaupun algoritma *Bee Colony-Tabu* memerlukan waktu pencarian solusi lebih lama, namun dengan penambahan waktu tersebut alternatif solusi yang didapatkan merupakan solusi yang berbeda daripada solusi yang didapatkan dari algoritma *Bee Colony*. Sehingga membuka kemungkinan urutan solusi yang lebih bervariasi daripada solusi dari algoritma *Bee Colony*.

Penelitian berikutnya disarankan untuk menganalisa pengaruh penggunaan *tabu list* dalam algoritma *metaheuristik* lainnya seperti *ant colony*, *particle swarm*, *cuckoo search*, atau *firefly algorithm*, untuk mengevaluasi hasil yang didapatkan pada penelitian ini.

DAFTAR PUSTAKA

1. Bedworth. David D. dan Bailey. James E., 1982, *Integrated Production Control System* , John Wiley and Sons, New York, Hal. 311-314.
2. Chong. C. S., Low. M. Y. H., Sivakumar. A. I., and Gay. K. L., 2006, "A bee colony optimization algorithm to job shop scheduling," in *Proc. of the 2006 Winter Simulation Conference, 2006*, pp. 1954–1961.
3. Chong. C. S., Low. M. Y. H., Sivakumar. A. I., and Gay. K. L., 2007, "Using a bee colony algorithm for neighborhood search in job shop scheduling problems," in *Proc. of 21st European Conference on Modeling and Simulation (ECMS2007).*
4. Goodman. E., Hedetniemi. S. T., 1977, *Introduction to the Design and Analysis of Algorithms*, McGraw-Hill.
5. Karaboga. S. and Basturk. D., 2007, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm", *Journal of Global Optimization, vol. 39*, no. 3, pp. 459–471.
6. Karaboga. S. and Basturk. D., 2008, "On the performance of artificial bee colony (abc) algorithm," *Applied Soft Computing, vol. 8, no. 1*, pp. 687–697.
7. Lucic. P. and Teodorovic. D., 2003, "Vehicle routing problem with uncertain demand at nodes: The bee system and fuzzy logic approach," in *Fuzzy Sets in Optimization*, J. L. Verdegay, Ed. Berlin / Heidelberg: Springer- Verlag, pp. 67–82.
8. Nakrani. S. and Tovey. C., 2004, "On honey bees and dynamic server allocation in internet hosting centers," *Adaptive Behavior, vol. 12*, no. 3-4, pp. 223–240
9. Nasution, Arman H., 2003, *Perencanaan dan Pengendalian Produksi* , Edisi Pertama. Surabaya: Guna Widya.
10. Nawaz, M., Enscore, E. E., and Ham, I.,. 1983. "A Heuristic Algorithm for the m-machine, n-job Flow Shop Sequencing Problem", *Omega*, No. 11, 91-95
11. Pham DT, Ghanbarzadeh A, Koc E, Otri S, Rahim S and Zaidi M. (2006). *The Bees*

Algorithm – A Novel Tool for Complex Optimisation Problems. Intelligent Systems Laboratory, Manufacturing Engineering Centre, Cardiff University, UK,

12. Seeley, T.D., S. Kühnholz, and A. Weidenmüller. 1996. The honey bee's tremble dance stimulates additional bees to function as nectar receivers. Behavioral Ecology and Sociobiology 39: 419-427
13. Schmidt, K. (May 18, 2001). *Using Tabu Search to Solve the Job Shop Scheduling Problem with Sequence Dependent Setup Times.*
14. Teodorovic. D., (2008), "Swarm intelligence systems for transportation engineering: Principles and applications," *Transportation Research Part C: Emerging Technologies, vol. 16*, no. 6, pp. 651–657.
15. Von Frisch. K., 1974, "Decoding the language of the bee," *Science, vol. 185*, no. 4152, pp. 663–668.
16. Watson, Barbulescu, Howe & Whitley, 1999, "Algorithm Performance and Problem Structurefor Flow-shop Scheduling", *American Association for Articial Intelligence*.